REMARKS

Favorable consideration of this application is respectfully requested.

Claims 1 - 11 are currently active in this case. Claims 1 - 8 have been amended by way of the present amendment. Each and amended claim is supported by the specification and claims as originally submitted and no new matter has been added.

In the outstanding Official Action, Claims 1-8 were objected to based on various informalities; and Claims 1 - 11 were rejected under 35 U.S.C. 103(a) as being unpatentable over *Heidorn, et al.* (U.S. Patent 5,966,686, hereinafter referred to as *Heidorn)*, in view of *Warren , et al.* (*Using Semantics in Non-context-Free Parsing of Montague Grammar, 1982*, hereinafter *Warren*) and in further view of *Ramaswamy, et al.* (US Patent 6,311,150, hereinafter referred to as *Ramaswamy)*.

Applicant appreciatively acknowledges the Examiner's suggested claim changes in Claims 1-8. Applicant has amended Claims 1-8 to correct the noted informalities as suggested by the Examiner.

Applicant respectfully traverses the rejection of Claim 1 under 35 USC 103(a) as being unpatentable over *Heidorn* in view of *Warren* and further in view of *Ramaswamy*. Claim 1 recites:

> ***1. (Currently Amended) A natural language processing apparatus for translating natural language into a formal language executable on a programmable device, said system comprising,***

a) memory for storing data;

b) a data processor;

c) an input device for presenting natural language text to said system:

d) a text parser for partitioning said text into a sequence pf sequences of strings of characters or pretokens;

e) a lexicon for storing lexical terms as token associated with lexical type and reference data;

f) a lexical type assignment process for assigning lexical types to pretokens by comparison to terms in the lexicon;

g) a lexical insertion processor for inserting terms into the lexicon under specific control;

h) a control processor for invoking lexical insertions under the condition that a pretoken is not recognized as a lexical token;

i) a type contextualization processor by which refined lexical types may be reassigned to tokens depending on syntactic context;

j) a type reduction matrix;

k) a term reduction processor which uses said type reduction matrix to determine proper syntactic dependencies between tokens in a sentence;

l) a term inversion processor for constructing chains of syntactic dependencies among lexical terms in an expression and for determining the proper dependencies between those chains;

m) a syntactic tree generation processor for constructing syntactic trees representing the syntactic structure of each processed expression;

n) a syntactic algebra comprising syntactic terms formally representing processed expressions;

o) a syntactic representation processor for constructing syntactic terms to represent the formal syntactic structure of processed expressions;

p) a semantic algebra comprising semantic objects as formal references of appropriate terms in the syntactic algebra;

q) a semantic representation processor for associating internal semantic object references with terms in the syntactic algebra;

r) a semantic tensor algebra comprising correlated pairs of syntactic algebraic terms and their semantic object representations;

s) a formal representation processor for associating appropriate internal formal models with terms in the semantic tensor algebra;

t) a formal interpretation processor for transforming terms in the syntactic algebra into equivalent expressions in an internal formal language;

u) an external representation processor for associating external operational environments with internal formal models;

v) an external interpretation processor for translating expressions in an internal in an internal formal language into equivalent formal expressions executable into appropriate external operational environments.

However, the cited references fail to teach or suggest similar subject matter.

Applicant respectfully traverses the assertion in the outstanding Office Action which states *"...Heidorn teaches a natural language processing apparatus for translating natural language into a formal executable on a programmable device..."* Applicant respectfully notes that *Heidorn* provides discussion regarding a method for generating a Logical Form Graph (LFG) from text. However, *Heidorn* does not teach how to create a formal executable from text; which is fundamentally different from *Heidorn's* LFGs and related processing.

Applicant respectfully notes that *Heidorn* mainly discusses the generation of a Logical Form Graph (LFG) from a Syntactic Parse Tree (SPT) (e.g., *Heidorn,* Abstract, line 6). Applicant respectfully notes that an LFG is a labeled, directed graph which is a semantic representation of an input sentence. Generated by a specific semantic analysis according to *Heidorn*, an LFG describes the meaning of a fragment of natural language text by exhibiting the sort of *"deep"* semantic structure not represented by an SPT alone. As a graph, an LFG has nodes and links, but is not hierarchically ordered. More particularly, the nodes of an LFG no longer represent the exact forms of input words, nor the exact relationships between them, i.e. there is no natural mapping from syntactic to semantic form, but instead a highly contrived relationship involving term insertions, deletions, and alterations according to a rule-based procedure. To quote directly from *Heidorn*: *"methods for generating logical form graphs involve computationally complex adjustments to, and manipulations of[,] the syntactic parse tree"*. *Heidorn* also discusses semantic rules as *"statements in a programming language"* that create a tree or graph node. *Heidorn*, col. 10, lines 22-27. However, neither LFGs nor programmed semantic rules describe or suggest a formal executable language from natural language.

In contrast, Claim 1 specifically recites translation of "... *natural language into a formal executable language."* Further, the claim language itself does not recite or generate anything like an LFG. Applicant respectfully notes that Claim 1 recites a computationally straightforward mapping from syntactic representations of text to semantic representations presented therein that is algebraically faithful, unlike the LFG processes and specific rules of Heidorn. In addition, none of the syntactic or semantic procedures presented in Claim 1 require the application of explicit rules as in Heidorn.

Applicant admits that *Heidorn* describes an NLP system comprising memory, data processor, text input device, and a text parser. However, those systems are not configured as stated in Applicant's Claim 1. More importantly, Applicant respectfully traverses any assertion that would equate Applicant's claimed *"type contextualization processor by which refined lexical types may be reassigned to tokens depending on syntactic context,"* to any portion of Heidorn. In particular, Applicant respectfully traverses the assertion that states that *Heidorn* reassigns *"whom"* due to context. In fact, *Heidorn* eliminates the word *"whom"* from his logical form graph as being unnecessary, *"...certain words that are unnecessary for conveying semantic meaning, such as 'The' and 'whom' do not appear in the* *logical form graph..."* (*Heidorn*, col. 5, lines 45 – 48, describing LFG's, a fundamental component of Heidorn; also note Phase III, col. 12, line 25 – col. 13, line 8, et al.). And, the disclosed invention does not apply rules to eliminate words or to infer meaning from external sources. Most important, *Heidorn* fails to teach or suggest a "*type contextualization processor by which refined lexical types may be reassigned to tokens depending on syntactic context."*

Warren and *Ramaswamy* similarly fail to teach or suggest a similar type contextualization. Accordingly, Applicant respectfully submits that the claimed invention cannot be obvious over *Heidorn* in view of *Warren* and *Ramaswamy* because the combined references fail to teach or suggest subject matter specifically claimed in Claim 1.

Further, as admitted in the outstanding Office Action, *Heidorn* does not teach Applicant's claimed control processor *"h) a control processor for invoking lexical insertions under the condition that a pretoken is not recognized as a lexical token;"* However, Official Notice was taken that the same would have been obvious.

Applicant respectfully traverses the Official Notice. In particular, Applicant respectfully notes that the motivation provided for the Official Notice was that it would have been obvious to add words to a dictionary that are not recognized. However, a lexical term is not simply a word but a word (token) AND an associated lexical type. And, it is not at all obvious how to insert such a pair into a dictionary because a paired word (token) and lexical type are unlike words in a standard dictionary, but Applicant's present disclosure teaches how to do the pairing and the insertion. Therefore, Applicants are entitled to broadly claim lexical insertions based on the stated criteria.

Warren and *Ramaswamy* also fail to teach or suggest lexical insertions under similar conditions. Accordingly, Applicant respectfully submits that the claimed invention cannot be obvious over *Heidorn* in view of *Warren* and *Ramaswamy* because the combined references fail to teach or suggest subject matter specifically claimed in Claim 1.

Regarding Applicant's claimed type reduction matrix and related portions (particularly claimed items j), k), and l)), Applicant respectfully traverses the assertion in the outstanding Office Action that states *Warren "... teach the type reduction matrix, the term reduction processor and the term inversion processor."* In particular, Applicant respectfully notes the cited passages from *Warren* (pg 134 c.1 para 5-c.2 para 2), and (pg 134, c.1 para 4-c.2 para 2). However, the cited passages do NOT teach any of the claimed matrix and related processes. For example, nowhere in the cited passages does *Warren* describe a type reduction matrix. In particular, *Warren* does not describe a method of reducing an ambiguous lexical type into a less ambiguous lexical type through the use of a type reduction matrix, a term reduction processor or a term inversion processor.

In fact, *Warren* provides no indication of lexical terms consisting of a token and a virtual lexical type. *Warren* 's reference to a matrix refers to a syntactic parse tree with multiple possible forms, and *Warren* then teaches how to combine semantics and syntactical rules to reduce the multiple possible forms to the most likely form. However, semantics or set syntactical rules to disambiguate the virtual lexical types associated with the tokens is not what is claimed by Applicant's claimed type reduction matrix, term reduction processor, and term inversion processor which perform reduction of an ambiguous lexical type into a less ambiguous lexical type.

Further, with respect to Applicant's claimed *"m) syntactic tree generation processor,"* Applicant respectfully traverses the assertion that states *Warren* teaches such a syntactic tree generation processor. Applicant admits that *Warren* constructs parse trees, in fact most NLPs will do so. However, as is the typical case, *Warren* constructs them without structure or relation to processed expressions. In contrast Claim 1 specifically recites construction of the tree so as to represent *"the syntactic structure of each processed expression."*

With respect to Applicant's claimed syntactic algebra n), o), p), and q), Applicant respectfully traverses the assertion that states *Warren* teaches the use of syntactic algebra, a syntactic representation processor, a semantic algebra and a semantic representation processor. In fact, the system discussed in *Warren* only processes grammars that are non-ambiguous. (e.g., *Warren*, pg. 124, col. 2, lines 3 – 5) *Warren* 's pairs are pairings of a disambiguated language (i.e., a non-ambiguous grammar) and a corresponding algebra (e.g., Warren, pg. 136, col. 2, lines 8 – 16).

In contrast, the present invention teaches and claims a practical method of disambiguating English text into a semantic object network. In this way, the present invention enables formal computer execution of English text, something that *Warren* (and/or any combination of *Warren* , *Heidorn*, and *Ramaswamy*) fail to teach or suggest. Therefore, Applicant respectfully submits that *Warren* fails to teach or suggest subject matter specifically claimed in Claim 1. Accordingly Applicant respectfully submits that Claim 1 is yet further patentable over the combination of Heidorn, *Warren* and *Ramaswamy*.

As further admitted in the outstanding Office Action, the combination of *Heidorn* and *Warren* fail to teach Applicant's claimed external applications u) and v). Applicant respectfully traverses the assertion in the outstanding Office Action that states *Ramaswamy* teaches *"a method for associating external operational environments with internal formal models."* In fact, *Ramaswamy* teaches how to associate his internal formal model with an existing command structure of a computer. However, the claimed invention associates lexical terms with operational terms, and, therefore, is different and also more flexible and useful than that described in *Ramaswamy*. The present invention includes disclosure of how to associate disambiguated lexical types with operational terms consisting of the token and the operational type (e.g., function, object, etc.). This provides a general mapping from English text to a formal executable language. In contrast, *Ramaswamy* is restricted to a specific set of computer-understood commands. Thus, the present invention teaches association at a much more granular token level, whereas *Ramaswamy* teaches how to associate at a macro command level in a preset and constrained language.

Therefore, Applicant respectfully submits that the cited references, individually or combined, not only fail to teach or suggest limitations specifically claimed in Claim 1, but that the general object of the claimed invention is also not addressed. Accordingly, Applicant respectfully submits that Claim 1 is patentable.

Applicant further respectfully notes that additional independent Claims 2-5, 10, and 11 are also distinguished over the combined references. As to Claim 2, Applicant respectfully traverses the assertion in the outstanding Office Action that equates Claim 2's steps a)-f), h)-m), and n)-p), steps of Claims 3-5, and Claims 10-11 on the same basis (same basis and same rationale) as being similar to limitations in Claim 1. Nevertheless, if the limitations are deemed similar, they are also distinguished for the same reasons that Claim 1 is distinguished over the cited art. Accordingly, Applicant respectfully submits that Claims 2-5, 10, and 11 are patentable over the cited art references.

Based on the patentability of the independent claims, Applicant further respectfully submits that dependent Claims 6-9 are also patentable.

Consequently, no further issues are believed to be outstanding, and it is respectfully submitted that this case is in condition for allowance. An early and favorable action is respectfully requested.

The Commissioner is hereby authorized to charge any fees (or credit any overpayment) associated with this communication and which may be required under 37 CFR §1.78 to Deposit Account No. 50-2603, **referencing Attorney Docket No. 358008.00100.**

Respectfully submitted,
REED SMITH LLP

Dated: 7 Nov 2005

By: [illegible]
Name: John W. Carpenter
Registration No. 39,129

Two Embarcadero Center
Suite 2000
PO Box 7936
San Francisco, CA 94120-7936
Direct Dial (415) 659-5927
(415) 543-8700 Telephone
(415) 391-8269 Facsimile